October 3, 2001
Q65312AP.CWB

AN APPARATUS, METHOD, AND COMPUTER PROGRAM FOR WIRE-SPEED CLASSIFICATION AND PRE-PROCESSING OF DATA PACKETS IN AN ATM NETWORK

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates generally to the monitoring and processing of Internet Protocol (IP) data transferred in an asynchronous transfer mode (ATM). More specifically, the present invention relates to the monitoring of data to determine if it complies with a certain set of rules for further processing, depending on the data flow classification, as the data is transmitted through an ATM switch. The application entitled "A Method and Apparatus for Wire-Speed Application Layer Classification of Data Packets" (U.S. Appln. No. 09/547,034) is assigned to a common assignee. The '034 application is herein incorporated by reference for all purposes.

2. Description of the Related Art

Data flows between a network of computers carrying portions of digital information between different nodes. Generally, the results of an application running at one network node may be sent to a computer at another network node. In order to establish the transfer of data, the information is encapsulated in data packets and transmitted over the network. Some

communication protocols transfer data packets in a half duplex mode, while others transfer data packets in a full duplex mode.

Two popular ways of transferring Internet Protocol (IP) data between network nodes are the asynchronous transfer mode (ATM) and the Ethernet packet format mode. Figure 1 describes a network 100 enabling the transfer of data from one Ethernet network 110 to another Ethernet network 110 through an ATM network 120. The Ethernet networks 110 are connected to the ATM network 120 via ATM gateways 130. However, the method used to transfer packets over the Ethernet networks 110 is different from the method used to transfer data over the ATM networks 120. One difference between the two methods is that the data packets (i.e. "ATM cells") in the ATM networks 120 have a fixed size, while the data packets (i.e. "Ethernet packets") in the Ethernet networks 110 vary in size. Another difference is that the ATM cells arrive at their destination in the same order in which they were transmitted from their source, while the Ethernet packets may arrive at their destination out of order.

The ATM is a communication technology designed to address long distance communication at high speeds with different networking systems connected at the end points. Unlike other communications protocols, the ATM transfers cells of data using fixed-length cells, each containing 53 bytes. As shown in Figure 2, an ATM cell 200 has a 5-byte header 205 and contains a 48-byte payload 210. By using cells 200 having a fixed size, transfer speeds are increased and delay variations are very low. This allows for a dependable

performance of the delivery system. The addressing system used for ATM, which was defined by standard committees, depends both on the end system and on the network node to which it is connected. Every network system or node is allocated a 13-byte network address, usually used by the routing protocols, to locate and find a path to a target. The header 205 of the cell 200 contains both a virtual channel identifier (VCI) 215 and a virtual path identifier (VPI) 220 to uniquely identify the channel and path of the cells 200 through the ATM network.

In contrast to the ATM networks 120, Ethernet networks 110 use a scheme based on IP addresses of the data packets to route payloads through a network in accordance with a full duplex protocol. The IP uses a unique identification for a process flow, also known as the IP tuple, which is shown in Figure 3. Specifically, the IP tuple 300 uniquely identifies a source of a data packet via a 4-byte source IP address 310 and a 2-byte source port 340. Also, the IP tuple 300 uniquely identifies a destination of the data packet by a 4-byte destination IP address 320 and a 2-byte destination port 350. In addition, a 1-byte protocol field 330 defines the protocol type used. Data is transferred over packetized networks, such as an Ethernet, by sending packets of data of variable sizes from a source to a destination, and the packets all have at least the tuple described for identification purposes.

In certain applications the tuple can be extracted from up to 64 bytes. Hence, there are cases where an IP tuple must be split between two ATM cells, as one ATM cell may carry a payload of no more than 48 bytes.

In an Ethernet network, data packets may be monitored for basic qualities in order to apply certain rules regarding such packets. For example, the IP tuple 300 of each data packet may be analyzed to determine the process flow to which it belongs, how the packet should be processed, where the packet should be routed, etc. The application of certain rules to certain data packets ensures a high quality of the transmission of real-time applications such as video or voice over Ethernet, avoids the transmission of restricted applications, and/or applies sets of other rules. However, as higher transmission speeds are required and the number of rules increases, it is essential to design systems that are efficient in handling the stream of packetized data transmitted through the system and that quickly and accurately apply rules to data packets. Since there is a common need to connect between ATM and packetized networks for the purpose of transferring data from one node to another in a mixed network, various ways have been proposed to accomplish this connection. One manner to more efficiently monitor the data in a mixed network is to monitor the IP data when flowing through an ATM node as part of an ATM cell.

While IP data can be classified for purposes of rule checking and enforcing actions by uniquely identifying its characteristics based on information contained in the header, it is essential to extract the header information from the ATM cells. The trivial approach would be to segment and reassemble (SAR) the IP data from the data in each cell. However, although this straightforward approach is simplistic, it requires the reassembly

of the entire IP data packet and/or IP tuple, and will degrade the wire-speed performance of the system.

SUMMARY OF THE INVENTION

In an illustrative, non-limiting embodiment of the invention, an apparatus that monitors data transported in ATM cells over an ATM communication network is provided. More specifically, the apparatus monitors IP packets transported over ATM networks.

In another illustrative, non-limiting embodiment of the invention, a method is provided, in which associated packets are recognized and grouped for further packet processing after a classification process, as well as the back annotation to the ATM cells. An important feature of this non-limiting embodiment is that the design allows for scaling the solution in order to efficiently address increasing traffic loads.

In another illustrative, non-limiting embodiment of the invention, a method is provided, in which load is balanced between packet processors, otherwise known as network processors, as well as the specific functionality of the data path and packet classifier units.

BRIEF DESCRIPTION OF THE DRAWINGS

Various aspects of non-limiting embodiments of the present invention will become more apparent by describing such embodiments below in conjunction with the attached drawings, in which:

Figure 1 is a schematic diagram of an example of a connection between Ethernet (packetized) and ATM networks;

Figure 2 is an example of an ATM cell format;

Figure 3 is an example of an IP tuple;

Figure 4 is a diagram of an illustrative embodiment of an ATM Network Monitoring and Classifying System;

Figure 5 is a diagram of an illustrative embodiment of a memory assignment scheme for tuple restructuring; and

Figure 6 is a flow chart for an illustrative embodiment of Tuple Extraction from an ATM Cell and Cross Reference.

DESCRIPTION OF THE ILLUSTRATIVE EMBODIMENTS OF THE INVENTION

The following description of the embodiments discloses specific configurations, features, and operations. However, the embodiments are merely examples of the present invention, and thus, the specific features described below are merely used to more easily describe such embodiments and to provide an overall understanding of the present invention. Accordingly, one skilled in the art will readily recognize that the present invention is not limited to the specific embodiments described below. Furthermore, the descriptions of various configurations, features, and operations of the present invention that would have been known to one skilled in the art are omitted for the sake of clarity and brevity.

An illustrative embodiment of the present invention may be employed in a policy-based network system shown in Figure 4. The system 400 is connected to an ATM network 120 through a Physical Access Unit 410, which

comprises standard ATM interface components and which captures the data flowing on the high speed ATM network in ATM cells. The cells are processed by a Data Path Unit 420 in conjunction with the operation of a Header Processor 430 and a Classifier 440. The Header Processor 430 and Classifier 440 evaluate each cell and determine, according to predefined rules, whether the operation on the cell should continue. The Data Path Unit 420 reconstructs an IP data packet from the data cells, and if necessary, provides the data to Packet Processors 450 for continued processing. An example of how the data is handled and its association with a process flow are described in detail in the '034 application mentioned above.

As will be described in more detail below, the Data Path Unit 420 and the Header Processor 430 may be employed to process data received in an ATM cell format rather than in an IP data packet format. Both the Data Path Unit 420 and Header Processor 430 are capable of ignoring cells that contain data other than IP data. One illustrative manner in which the Data Path Unit 420 and the Header Processor 430 determine whether or not the data in the cell is IP data is to examine the payload type identifier 230 contained within each header 205 of each cell. (See Fig. 2).

In one implementation of the present embodiment, the Header Processor 430 performs wire-speed assembly of the IP tuple 300 from the ATM cell 200 to determine the remaining operations, if any, to be performed on the cells 200 belonging to an IP packet. One illustrative way to perform such assembly is to allocate certain memory space. For example, as shown in

Figure 5, three portions of memory are provided: Pointer Memory ("PM") 520, Cell Information Memory ("CIM") 530, and Flow Information Memory ("FIM") 540. A pointer 510, which is constructed from the VCI/VPI data 215 and 220 from the ATM cell 200 points to a location 525 in the PM 520 which is unique to the combined value of the VCI/VPI data 215 and 220. The location 525 in the PM 520 contains two fields of information. One field is a validity status field, corresponding to the pointer 510, and the other field is a pointer field that points to another memory location, depending on the content of the validity status field V. The field V may have the following values:

00 – invalid pointer

01 – saved for future use

10 – cell pointer

11 – flow pointer

When the validity status field V has the value "00," the data contained in the pointer field may not be used as a pointer and is useless information. When the field V has the value "10," the pointer is used to point to the CIM 530, where the content of the current ATM cell 200 is stored as cell information 535. The storage of the current cell 200 is necessary when the cell 200 does not contain a full IP tuple. When the field V has the value "11," the pointer is used to point to the FIM 540, where the information of the process flow is stored. The value of the field V remains valid until the last cell 200 of the data packet is received. Once the last cell 200 of the data packet is received, as indicated in the cell header 205, the value of the field V is

invalidated by resetting it to "00". Failure to reset the field to "00" may result in VCI/VPI data 215 and 220 (i.e. the pointer 510) pointing to the wrong process flow information. However, it is guaranteed that all the cells 200 with the same VCI/VPI data 215 and 220 between the first and last cell 200 all arrive in sequence and all belong to the same data packet. It should be noted that it is possible that cells containing packets with different VCI/VPI addresses 215 and 220 may be flowing through the system at the same time.

An illustrative, non-limiting embodiment of a method of the present invention will now be described.

A current cell 200 is received by the Header Processor 430 as shown in operation 610 of Figure 6. The cell 200 is checked in operation 620 to determine if it contains a full IP tuple 300, or if it is necessary to wait for the next cell 200 having the same VCI/VPI data 215 and 220 to obtain the rest of the IP tuple 300. If the current cell 200 does not contain a full IP tuple 300, then in operation 630, the payload of the current cell 200 is stored in cell information 535 of the CIM 530, and a pointer 525 in the PM 520 to the cell information 535 is created. The pointer comprises a validity status field V, which equals "10", and a pointer field, which contains the address in the CIM 530 where the cell information 535 is located. In one implementation, the cell information 535 includes the payload 210 of the first cell.

In another illustrative, non-limiting embodiment of the present invention the entire content of the current cell 200 is saved in the CIM 530 at the location pointed to by the pointer field in the PM 520. In operation 640,

the Header Processor 430 receives the next cell that contains the second part of the IP tuple 300.

If two cells 200 were necessary to create the full IP tuple 300, (i. e. if operations 630 and 640 were executed) then, in operation 650, the information previously stored in the CIM 530 as the cell information 535 is used in conjunction with the second cell payload 210 to reconstruct the full IP tuple 300. On the other hand, if the IP tuple 300 was contained within the first cell, (i.e. if operations 630 and 640 were not executed) the IP tuple 300 is extracted from the current cell 200.

The IP tuple 300 is then checked to identify whether or not it belongs to a process flow that has already been designated by a process flow identifier (operation 660). If the IP tuple 300 is part of a known flow, the Classifier 440 returns the flow information in operation 670, which includes the Flow-ID, the Packet Processor number and other control/status information. This information is required for the later packet processing and is stored in the FIM 540 as flow information 545. If the IP tuple 300 corresponds to the first data packet of a new process flow, a new process flow entry is generated during operation 680. The information is stored in the FIM 540, and a pointer is created to the stored information during operation 690. The pointer comprises a validity status field V, which equals "11" and a pointer field which contains a pointer to the beginning of the flow information 545. When identified as belonging to a certain flow, the data packet corresponding to the tuple 300 is scheduled to be processed via a designated Packet Processor from the

available Packet Processors 450. Then, the data packet is made available to the designated processor via the Data Path Unit 420. The symmetrical and balanced architecture of the system 400 allows for additional Packet Processors 450 to be easily added in order to increase processing bandwidth and, hence, the performance of the entire system 400.

The Data Path Unit 420 operates at full wire speed. Therefore, the Data Path Unit 420 assists the Header Processor 430 in providing indications of how to construct the cells 200, and uses the information provided by the Classifier 440 with respect to the process flow affiliation of a reassembled packet. Moreover, it is beneficial for the overall system performance to ignore the cells 200 that require no reassembly, because the cells 200 that do not require reassembly may contain IP data packets. According to the system rules, these IP data packets may not require any processing, and, hence, it would be wasteful to reassemble them. Therefore, the Header Processor 430 and/or the Classifier 440 may generate commands to the Data Path Unit 420 with instructions on how to handle cells having a certain VCI/VPI data 215 and 220.

Once reassembled, the Data Path Unit 420 performs several consistency checks on each packet, including IP and TCP checksums, IPV4, and legality of packet length. The interface between the Data Path Unit 420 and the plurality of Packet Processors 450 provides the required information for further packet processing.

The operations shown in Fig. 6 may be implemented by software

which is executed by the Header Processor 430 shown in Fig. 4. Also, other components shown in Fig. 4 may alternatively or additionally perform some or all of the operations shown in Fig. 6, as well as other operations. Also, the software may be supplied to the Header Processor 430 and/or other components via a read only memory ("ROM"), a random access memory ("RAM"), a floppy disk, a hard disk, an optical disk, a carrier wave (e.g. a carrier wave transmitted via the internet, a vertical blanking interval of a television signal, etc.), or any other computer readable medium.

Although the preferred embodiments of the present invention have been described, it will be understood by those skilled in the art that the present invention should not be limited to the described preferred embodiments, but various changes and modifications can be made within the spirit and scope of the present invention as defined by the appended claims.